

1. Get Post (Unauthenticated - HTTPS) https://127.0.0.1:8000/api/posts/

GET 1. Get Post (Unauthentic

No environment

Security (HTTPS) / 1. Get Post (Unauthenticated - HTTPS)

Save

Share

GET

https://127.0.0.1:8000/api/posts/

Send

Docs

Params

Authorization

Headers (6)

Body

Scripts

Settings

Cookies

Query Params

	Key	Value	Description		Bulk Edit
	Key	Value	Description		

Body

Cookies

Headers (12)

Test Results

200 OK

28 ms

421 B

Save Response

{ } JSON

Preview

Visualize

1

[ ]

## 2. Get Users (Unauthenticated - HTTPS) https://127.0.0.1:8000/api/users/

Security (HTTPS)

GET 2. Get Users (Unauthent

+

No environment

Security (HTTPS) / 2. Get Users (Unauthenticated - HTTPS)

Save

Share

GET

https://127.0.0.1:8000/api/users/

Send

Docs

Params

Authorization

Headers (6)

Body

Scripts

Settings

Cookies

Query Params

	Key	Value	Description		Bulk Edit
	Key	Value	Description		

Body

Cookies

Headers (12)

Test Results

200 OK

14 ms

524 B

Save Response

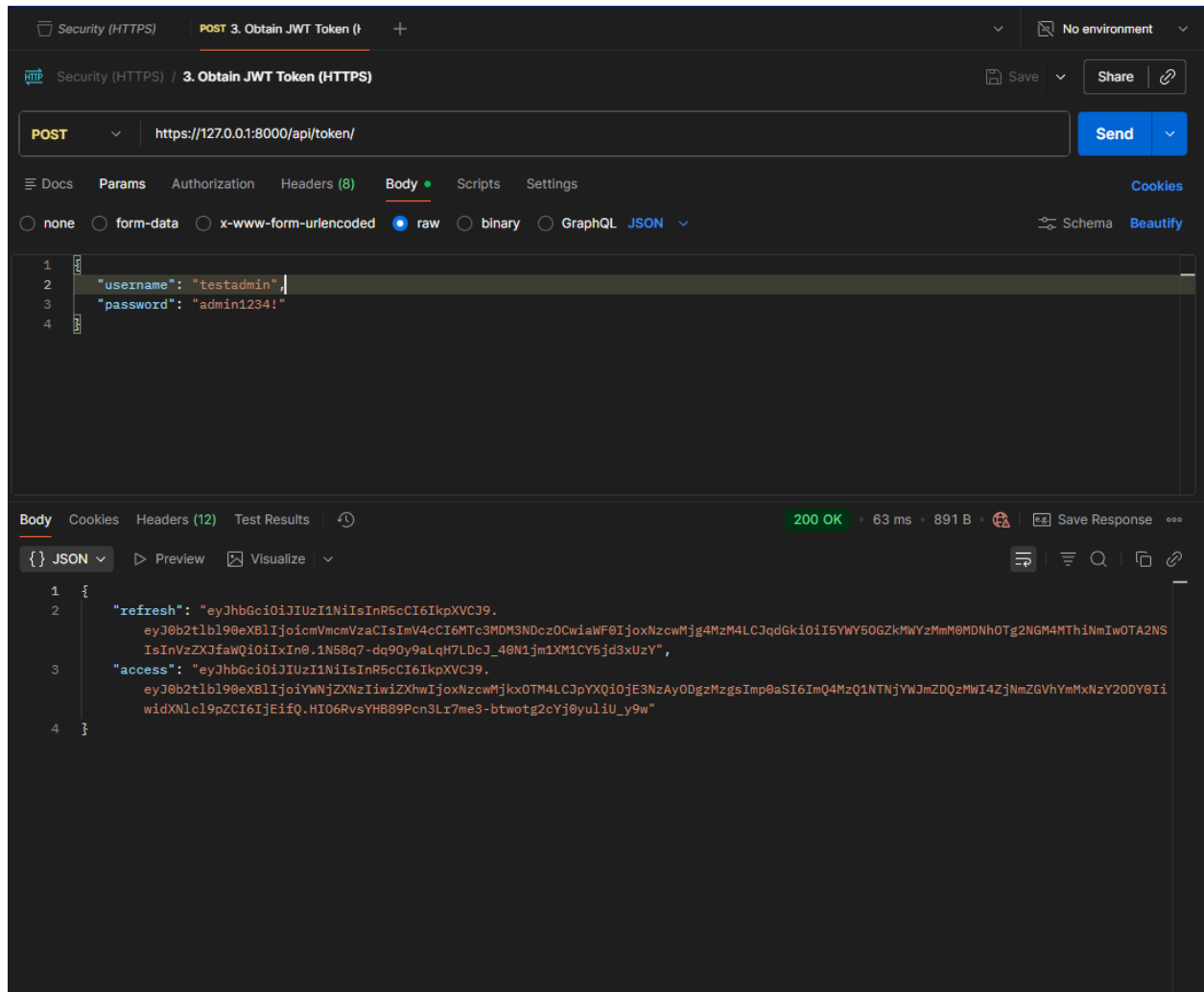
JSON

Preview

Visualize

```
1  [
2    {
3      "id": 1,
4      "username": "testadmin",
5      "email": "test@example.com",
6      "created_at": "2026-02-05T10:13:49.062009Z"
7    }
8  ]
```

### 3. Obtain JWT Token (HTTPS) <https://127.0.0.1:8000/api/token/>



#### 4. Create User (Authenticated - HTTPS) <https://127.0.0.1:8000/api/users/>

The screenshot displays a REST client interface with the following details:

- Request:**
  - Method: **POST**
  - URL: `https://127.0.0.1:8000/api/users/`
  - Body (raw):

```
1 {
2   "username": "testuser1",
3   "email": "testuser1@example.com",
4   "password": "securepassword123"
5 }
```
- Response:**
  - Status: **201 Created**
  - Time: 58 ms
  - Size: 524 B
  - Body (JSON):

```
1 {
2   "id": 2,
3   "username": "testuser1",
4   "email": "testuser1@example.com",
5   "created_at": "2026-02-05T10:49:53.722266Z"
6 }
```

## 5. Create Post (Authenticated by Author - HTTPS) <https://127.0.0.1:8000/api/posts/>

The screenshot displays a REST client interface with a dark theme. At the top, a tab is labeled "POST 5. Create Post (Author)". The URL bar shows "https://127.0.0.1:8000/api/posts/" with a "Send" button to its right. Below the URL bar, the "Body" tab is selected, showing a raw JSON request: 

```
1 {
2   "content": "My second authenticated post!"
3 }
```

 The bottom panel shows the response, with a status bar indicating "201 Created" in green, along with "24 ms" and "541 B". The response body is displayed in JSON format: 

```
1 {
2   "id": 1,
3   "content": "My second authenticated post!",
4   "author_username": "testadmin",
5   "created_at": "2026-02-05T10:52:05.839454Z"
6 }
```

## 6. Update Post (Authenticated by Author - HTTPS) <https://127.0.0.1:8000/api/posts/2/>

The screenshot shows a REST client interface with the following details:

- Request Method:** PUT
- URL:** `https://127.0.0.1:8000/api/posts/1/`
- Body:** A JSON object with one key: `"content": "Updated content for my second secure post"`
- Response:** A 200 OK status with a response body containing: `{ "id": 1, "content": "Updated content for my second secure post", "author_username": "testadmin", "created_at": "2026-02-05T10:52:05.839454Z" }`

The interface includes tabs for Docs, Params, Authorization, Headers (9), Body, Scripts, and Settings. The Body tab is active, showing the request and response JSON. The response status is 200 OK, with a response time of 15 ms and a size of 562 B.

7. Delete Post (Authenticated by Author - HTTPS) <https://127.0.0.1:8000/api/posts/2/>

Security (HTTPS)

DEL 7. Delete Post (Authenti

+

No environment

Security (HTTPS) / 7. Delete Post (Authenticated by Author - HTTPS)

Save

Share

DELETE

https://127.0.0.1:8000/api/posts/1/?

Send

Docs

Params

Authorization

Headers (7)

Body

Scripts

Settings

Cookies

none

form-data

x-www-form-urlencoded

raw

binary

GraphQL

JSON

Schema

Beautify

1

Ctrl+Alt+P to Ask AI

Body

Cookies

Headers (11)

Test Results

204 No Content

14 ms

401 B

Save Response

Raw

Preview

Visualize

1

8. Create Regular User (Authenticated - HTTPS) <https://127.0.0.1:8000/api/users/>

POST 8. Create Regular User

Security (HTTPS) / 8. Create Regular User (Authenticated - HTTPS)

POSThttps://127.0.0.1:8000/api/users/

Send

DocsParamsAuthorizationHeaders (9)BodyScriptsSettings

noneform-datax-www-form-urlencodedrawbinaryGraphQLJSON

```
1 {
2   "username": "regularuser",
3   "email": "regular@example.com",
4   "password": "regularpassword"
5 }
```

BodyCookiesHeaders (12)Test Results

201 Created • 12 ms • 524 B • Save Response

JSONPreviewVisualize

```
1 {
2   "id": 4,
3   "username": "regularuser",
4   "email": "regular@example.com",
5   "created_at": "2026-02-05T11:03:42.922234Z"
6 }
```



## 9. Obtain JWT Token (Regular User - HTTPS) <https://127.0.0.1:8000/api/token/>

The screenshot displays a REST client interface for a POST request. The request is sent to `https://127.0.0.1:8000/api/token/` with a JSON body containing `username: "regularuser2"` and `password: "regularpassword2"`. The response is a 200 OK status with a response time of 111 ms and a body size of 891 B. The response body is a JSON object containing `refresh` and `access` tokens.

**Request:**

```
POST https://127.0.0.1:8000/api/token/

{
  "username": "regularuser2",
  "password": "regularpassword2"
}
```

**Response:**

```
200 OK • 111 ms • 891 B

{
  "refresh": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ0b2t1b190eXB1IjoicmVmcmVzaCI6ImV4cCI6MTc3MDM3NjIjXMSwIiwiaWF0IjoxNzcxMjg5ODExLCJqdGkiOiI5ODh0DAwNmJkMzY0YTVhOGNhMzI4OTQ4Yzc4ZmM0YyIsInVzZXJfaWQiOiI1In0.I1jQ4ML2WBTWl2kXe7PXFIMHZdbiUJk8rTqhKxJ8Ms8",
  "access": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ0b2t1b190eXB1IjoicmVmcmVzaCI6ImV4cCI6MTc3MDM3NjIjXMSwIiwiaWF0IjoxNzcxMjg5ODExLCJqdGkiOiI5ODh0DAwNmJkMzY0YTVhOGNhMzI4OTQ4Yzc4ZmM0YyIsInVzZXJfaWQiOiI1In0.I1jQ4ML2WBTWl2kXe7PXFIMHZdbiUJk8rTqhKxJ8Ms8"
}
```

## 10. Regular User Create Post (HTTPS) <https://127.0.0.1:8000/api/posts/>

The screenshot displays a REST client interface with the following details:

- Request:**
  - Method: **POST**
  - URL: `https://127.0.0.1:8000/api/posts/`
  - Body (raw):

```
1 {
2   "content": "Post by regular user 2"
3 }
```
- Response:**
  - Status: **201 Created**
  - Time: 27 ms
  - Size: 537 B
  - Body (JSON):

```
1 {
2   "id": 2,
3   "content": "Post by regular user 2",
4   "author_username": "regularuser2",
5   "created_at": "2026-02-05T11:14:08.381411Z"
6 }
```

## 11. Superuser Update Regular User Post (RBAC Fails - HTTPS)

<https://127.0.0.1:8000/api/posts/2/>

The screenshot shows a REST client interface with a PUT request to `https://127.0.0.1:8000/api/posts/2/`. The request body is a JSON object: `{ "content": "Superuser trying to change regular user's post" }`. The response is a `403 Forbidden` status with a response time of 8 ms and a body size of 496 B. The response body is a JSON object: `{ "detail": "You do not have permission to perform this action." }`.

PUT 11. Superuser Update Re +

Security (HTTPS) / 11. Superuser Update Regular User Post (RBAC Fails - HTTPS)

PUT `https://127.0.0.1:8000/api/posts/2/` Send

Docs Params Authorization Headers (9) Body Scripts Settings Cookies

none form-data x-www-form-urlencoded raw binary GraphQL JSON

```
1 {
2   "content": "Superuser trying to change regular user's post"
3 }
```

Body Cookies Headers (12) Test Results 403 Forbidden 8 ms 496 B Save Response

{ JSON Preview Try with different auth credentials

```
1 {
2   "detail": "You do not have permission to perform this action."
3 }
```

## 12. Regular User Update Own Post (RBAC Succeeds - HTTPS)

<https://127.0.0.1:8000/api/posts/>

The screenshot displays a REST client interface with the following details:

- Request:**
  - Method: PUT
  - URL: `https://127.0.0.1:8000/api/posts/2/`
  - Body (raw):

```
1 {  
2   "content": "Regular user updates their own post"  
3 }
```
- Response:**
  - Status: 200 OK
  - Time: 23 ms
  - Size: 559 B
  - Body (JSON):

```
1 {  
2   "id": 2,  
3   "content": "Regular user updates their own post",  
4   "author_username": "regularuser2",  
5   "created_at": "2026-02-05T11:14:08.381411Z"  
6 }
```

### 13. Create Post (Unauthenticated Attempt - HTTPS) <https://127.0.0.1:8000/api/posts/>

The screenshot shows a REST client interface with a dark theme. At the top, the title bar reads "Security (HTTPS) POST 13. Create Post (Unauth)". Below this, the breadcrumb "Security (HTTPS) / 13. Create Post (Unauthenticated Attempt - HTTPS)" is visible. The main area is divided into two sections: the request and the response.

**Request Section:**

- Method: **POST**
- URL: <https://127.0.0.1:8000/api/posts/>
- Body Type: **raw** (selected)
- Body Content:

```
1 {  
2   "content": "Attempt to post without a token."  
3 }
```

**Response Section:**

- Status: **401 Unauthorized** (6 ms, 526 B)
- Body Type: **JSON** (selected)
- Body Content:

```
1 {  
2   "detail": "Authentication credentials were not provided."  
3 }
```

The interface includes various tabs like "Docs", "Params", "Authorization", "Headers (8)", "Body", "Scripts", "Settings", "Cookies", "Test Results", and "Pass the correct auth credentials". There are also buttons for "Save", "Share", "Send", "Schema", and "Beautify".